Dynamic Class Prediction with Classifier Based Distance Measure

Şenay Yaşar Sağlam[1] **W. Nick Street**[2]

[1] University of Iowa,
108 John Pappajohn Business Building,
Iowa City, IA 52242, U.S.A.
Email: senay-yasarsaglam@uiowa.edu

[2] University of Iowa,
108 John Pappajohn Business Building,
Iowa City, IA 52242, U.S.A.
Email: nick-street@uiowa.edu

Abstract

Combining multiple classifiers (ensemble of classifiers) to make predictions for new instances has shown to outperform a single classifier. As opposed to using the same ensemble for all data instances, recent studies have focused on dynamic ensembles in which a new ensemble is chosen from a pool of classifiers specifically for every new data instance. We propose a system for dynamic class prediction based on a new distance measure to evaluate the distance among data instances. We first map data instances into a space defined by the class probability estimates from a pool of two-class classifiers. We dynamically pick classifiers (features) to be used and the k-nearest neighbors of a new instance by minimizing the distance between the neighbors and that instance in a two-step framework. Results of our experiments show that our measure is effective for finding similar instances and our framework helps making more accurate predictions.

Keywords: Classification, Dynamic Ensembles, Confidence, Probability Estimates, Distance Measures, k-NN.

1 Introduction

An ensemble of classifiers consists of a set of trained classifiers whose individual decisions are combined to classify new instances. Most existing methods construct static ensembles, in which only one ensemble is chosen from the pool and used for all new instances. Recently, there have been studies in which each new data instance is treated individually. Since different instances are often associated with different classification difficulties, it is hypothesized that using different classifiers for the classification task rather than a single static ensemble of classifiers can improve performance. In this study, we propose a method to make dynamic predictions to address the following questions:

- To find k-nearest neighbors of a new instance, should original feature space or classifier space be used?
- Is using class probability estimates better than just classifiers' predictions to find similar instances?
- Is classifiers' performance on the validation instances helpful when finding similar instances?
- Should all of the classifiers in the pool be used for computing similarity between instances? If not, which criteria should be considered to eliminate certain classifiers?
- After neighbors are found, should we use them to form an ensemble or to make a prediction?

In Section 2, we summarize the earlier studies regarding ensemble of classifiers. In Sections 3 and 4, the general structure of the system and its running time are explained. In Sections 5 and 6, we explain proposed and baseline distance measures. We carry out several experiments to evaluate performance of our framework. The results are presented in Section 7. Finally, Section 8 concludes the paper.

2 Literature Review

Ensembles have received great attention over the past two decades because averaging errors of multiple predictors increases overall accuracy. Ensemble generation methods can use sampling to create different training sets, use different training parameters for the learning algorithm, or employ different learning algorithms on the same training set to generate a diverse set of classifiers (as there is no point in having the same classifier multiple times). Some well-known algorithms for creating ensembles include *Bagging* (Breiman (1996)), *Boosting* (Freund & Schapire (1996)), *Random Subspace Method* (Ho (1998)), and *Stacking* (Wolpert (1992)). Recently, *Overproduce and Choose strategy*, in which a large initial pool of candidate classifiers is produced and then a classifier or a subset of classifiers is selected, has been adopted in several studies (Didaci & Giacinto (2004), Dos Santos et al. (2008), Giacinto & Roli (2001), Ko et al. (2008), Woods et al. (1997), Zhang et al. (2006)).

Several factors contribute to the generalization ability of an ensemble. The first is the accuracy of the base classifiers. It is easy to verify that constructing an ensemble with the most accurate classifiers will result in good generalization error. However, since the classifiers require uncorrelated errors for the ensemble to be effective, diversity among them is also important. In other words, combining accurate and diverse classifiers may boost the performance of an ensemble if they have uncorrelated errors (Tumer & Ghosh (1996)). Meanwhile, posterior class probability estimates returned by classifiers have also been considered. For example, confidence measures based on ensembles' vote margin is proposed by Dos Santos et al. (2008) to improve the quality of their solutions or to be used as an input to a meta classifier in Stacking (Menahem et al. (2009), Ting & Witten (1997)).

Earlier studies regarding this dynamic scheme focus on selecting a classifier based on different features or different regions of the instances, depending on the similarities among them (Didaci et al. (2005), Didaci & Giacinto (2004), Giacinto & Roli (2001), Woods et al. (1997)). In dynamic *classifier* selection, a single predictor is chosen based on its likelihood to correctly predict the test pattern (Didaci et al. (2005)). For example, dynamic classifier selection approaches based on estimating local competence in selection regions defined by k-nearest neighbors, also known as the *k-Nearest Neighbor (k-NN) method*, have been proposed by Didaci et al. (2005), Didaci & Giacinto (2004), Giacinto & Roli (2001), Woods et al. (1997). However, unlike the k-NN algorithm, which uses the prediction values from k instances that are "closest" or "most similar" to the new data instance, the prediction of the most competent classifier in the region is used to make the decision.

Similar to the static classifier selection methods, the drawback of dynamic classifier selection methods is that the choice of a single individual classifier over the rest depends on how much we trust in that classifier's performance. If that classifier makes an incorrect decision, we will not be able to correct that decision. Therefore, a dynamic ensemble selection approach has been proposed by later studies (Dos Santos et al. (2008), Ko et al. (2008), Cavalin et al. (2010)). These studies focus on dynamic ensemble selection rather than a single classifier selection to overcome this drawback. Dos Santos et al. populate a set of candidate ensembles from a large initial pool of candidate classifiers, then choose an ensemble from that set for each new data instance based on candidate ensembles' confidence on this instance. Dos Santos et al. define confidence as a measure of the ensemble, based on vote margin, and use it as a second-level objective after optimizing on accuracy and diversity.

Ko et al. propose a method which, for each new data instance, finds its nearest k neighbors in the validation set using the original feature set, and dynamically chooses an ensemble based on the estimated accuracy of the classifiers in this local region of the new data instance. Two different versions of this method are proposed by Ko et al. (2008) in terms of how classifiers are chosen to form an ensemble for the new instance: KNORA-Eliminate and KNORA-Union. KNORA-Eliminate uses the classifiers that correctly classify every instance in the local region of the new instance. However, in KNORA-Union method classifiers that correctly classify at least one of the neighbors are added to the ensemble and each classifier submits a vote for each neighbor it classifies correctly to predict the new instance. Following this approach, Cavalin et al. adopt a hybrid framework and employ the confidence measure defined by Dos Santos et al. (2008) to build the ensembles. However, when the confidence value is not enough, Cavalin et al. search for the "closest" or "most similar" instance to the new instance in the validation set and assign its label to the new instance. Similarity measures are defined based on assigned class labels. We agree that similarity should be based on base classifiers' outputs as we will not gain much by using the same information (same feature set) used to train the classifiers. However, using predictions of the classifiers alone will not provide enough information to accurately define similarity between points. The KNORA method (Ko et al. (2008)) has been improved by Vriesmann et al. (2012). To specify the local region of a new data point, Vriesmann et al. investigate the effects of using a different distance measure on accuracy and conclude that the choice of distance measure has no effect on the performance of KNORA. Furthermore, different strategies for combining the information obtained from k neighbors of the new instance and the output of KNORA are adopted. Based on the experimental results, Vriesmann et al. suggest that additional information provided by the k-NN improves the performance of KNORA.

With KNORA, Ko et al. indicate that using neighbor information of a new data point even for constructing an ensemble can prove useful. However, it has been shown that using neighbors of a new data instance not just for constructing the ensemble but in fusion with that ensemble's decision for that instance enhances the performance (Cavalin et al. (2010), Vriesmann et al. (2012)). In particular, Vriesmann et al. find the neighbors of a new data instance based on the original feature space that is also used to train classifiers in the pool. This indeed duplicates the information at hand and increases the performance of the system to a certain point. Cavalin et al. address this issue by defining the similarity between instances based on classifiers' prediction on the validation and new data instances. However, classifiers' probability estimates are more informative compared to using the predictions alone. This is because they provide information on not only whether the classifier assigns the same labels to these instances, but also how confident it is with its decisions.

3 Proposed Framework

The prediction probability returned by a classifier can be considered as a measure of proximity of a data instance to the decision boundary. This measure can be used to compare multiple instances and to decide whether that particular classifier considers those data instances similar. Our proposed framework consists of two main steps: a static step and a dynamic step. We illustrate both steps as a flow chart in Figure 1.

In the static step (on the left in Figure 1), we have a pool of classifiers, $\mathcal{C} = C_1, ..., C_N$, of size N, and we map data instances into a new space defined by the class probability estimates from each classifier for

a given class label ℓ_1. Since we consider two-class problems, the choice of class label does not change our results. Next, we find each new instance's k-nearest neighbors in the validation set $\mathcal{V}$, of size M, using this space. These k neighbors, denoted by the set $\mathcal{V}'$, can be considered as the data instances that, on average, the classifiers agree are similar to the new instance. This step is referred to as "static" because, for all new instances, we consider the output of all N classifiers in the pool.

In the dynamic step of our framework (on the right in Figure 1), we use the results from the static step to select a subset of classifiers $\mathcal{C}'$, of size $E < N$, from the original pool, $\mathcal{C}$, that are suitable for classifying a given new instance. Reducing the size of the space is important because the number of classifiers in our framework is large and (dis)similarity becomes less meaningful as the feature space dimensionality increases. Our selection method favors classifiers that have high confidence in their predictions for the neighbors identified in the static step (i.e. classifiers for which the predictions are away from the 0.5 decision boundary).[1] Confidence of a classifier C_n on k neighbors is calculated as follows:

$$Conf_{C_n} = \frac{\sum_{i=1}^{k} \Pr(o_{i,n}|i)}{k}$$

where $o_{i,n}$ represents the label assigned to instance i by classifier C_n and $\Pr(o_{i,n}|i)$ represents the probability estimate returned by the classifier for the assigned class label. In other words, the confidence of a classifier on k neighbors is the average of probability estimates returned for the decisions. Most confident E classifiers are chosen to form the reduced space. Reducing the size of the classifier set in this manner avoids the unstable behavior that may occur near the decision boundary of some classifiers.

Once the classifier subset, $\mathcal{C}'$, is generated, the original distance measure is reapplied to find a new set of neighbors $\mathcal{V}^*$ for the new instance. This neighborhood is then used for the final classification of the instance; the new instance is assigned to the class label most common among its k neighbors.

4 Running Time Analysis

To analyze the running time of the system, we should first evaluate the static and dynamic steps separately. The running time of the static step can be analyzed in two parts: (1) forming the probability space; and (2) finding k neighbors of a new instance. Forming the probability space consists of getting predictions for M validation instances from N classifiers which takes $O(M \times N)$. This is a preprocessing step as once this space is formed, it will remain same for all new data instances. Finding k neighbors requires getting prediction for each new instance from N classifiers ($O(N)$), calculating distance between the new instance and validation instances ($O(M \times N)$), and finally finding k instances that are closest to the new instance ($O(k \times M)$). Overall, the running time for this static step is $O(M \times N)$ which is for calculating the distance between a new instance and the validation instance.

The dynamic step is composed of two main parts: (1) reducing the classifier space; and (2) finding new k neighbors of a new instance. Since predictions for validation instances and k neighbors of a new instance are already found in the static step, the running time of reducing the space consists of calculating the confidence of classifiers on the neighbors ($O(k \times N)$) and finding the most confident E classifiers ($O(E \times N)$). The running time of the rest of the dynamic step can be analyzed in a similar manner to the static step ($O(k \times E)$). The overall running time of the whole system is dominated by the running time of the static step as $k < M$ and $E < N$.

5 Baseline Distance Measures

In this section, we consider three different distance measures for our baseline: Euclidean Distance, Template Matching, and Oracle-Based Template Matching. Two of these measures are proposed by Cavalin et al. (2010) in further detail.

5.1 Euclidean Distance

We first consider the Euclidean Distance (ED) between two data instances, i and j, which can be expressed as:

$$ED_{i,j} = \sqrt{\frac{\sum_{d=1}^{D} \left(\phi_{i,j,F_d}\right)^2}{D}}$$
$$\phi_{i,j,F_d} = F_d(i) - F_d(j)$$

where a data set consists of the feature set $\mathcal{F} = \{F_d | d = 1, \ldots, D\}$ and $F_d(i)$ represents the value of feature F_d for data instance i.

5.2 Template Matching

Given a set of classifiers $\mathcal{C} = \{C_n | n = 1, \ldots, N\}$, Template Matching (TM) considers the percentage of classifiers, denoted by $TM_{i,j}$, that agree on the label of test instance i and validation instance j:

$$TM_{i,j} = \frac{\sum_{n=1}^{N} \alpha_{i,j,n}}{N}$$
$$\alpha_{i,j,n} = \begin{cases} 1, & \text{if } o_{i,n} = o_{j,n} \\ 0, & \text{otherwise.} \end{cases}$$

where $o_{j,n}$ represents the label assigned to instance j by classifier C_n and $\alpha_{i,j,n}$ represents whether classifier C_n assigns the same label to instances i and j. It follows that the higher $TM_{i,j}$, the more similar the pair of instances (i, j).

5.3 Oracle-Based Template Matching

Oracle-based Template Matching (OTM) expands upon TM by taking into account the correctness of classifiers on data instances. Specifically, when evaluating the similarity of test instance i to validation instance j

[1] For completeness, other selection methods are considered in Section 7.7.

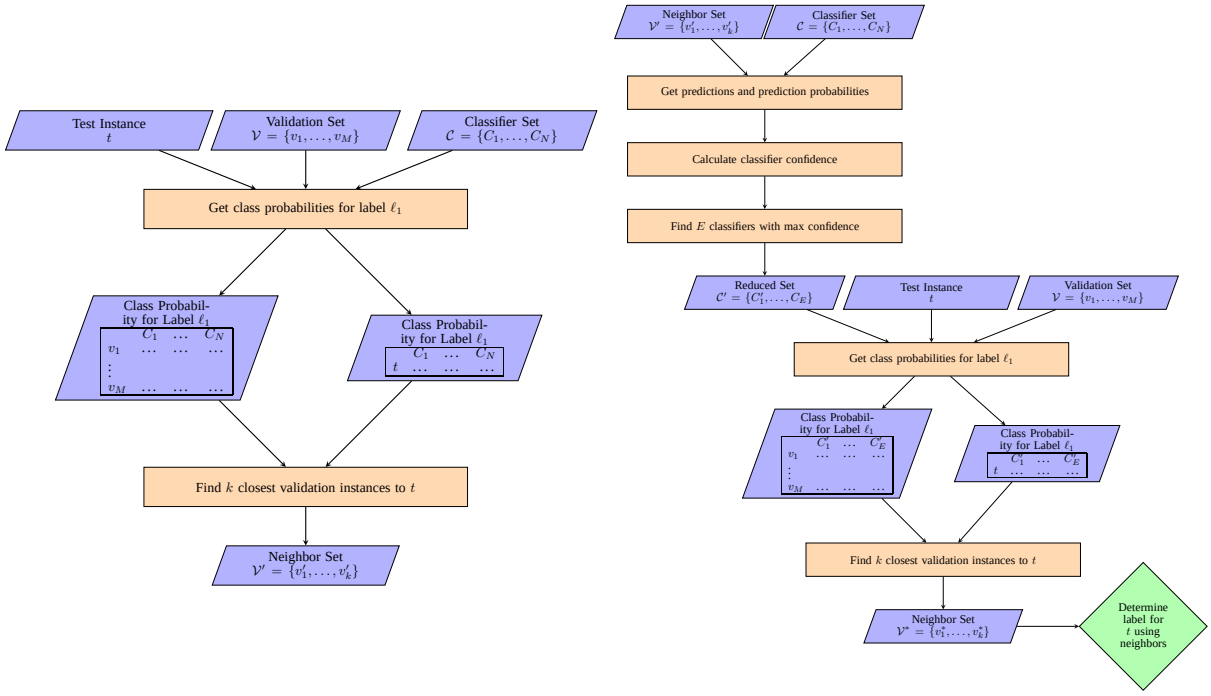


Figure 1: An overview of the proposed framework: Static Step (on the left) and Dynamic Step (on the right)

using OTM, only those classifiers that correctly classify j are considered. This measure is represented by the quantity $OTM_{i,j}$ as follows:

$$OTM_{i,j} = \frac{\sum_{n=1}^{N} \beta_{i,j,C_n}}{\sum_{n=1}^{N} \gamma_{i,j}}$$

$$\beta_{i,j,C_n} = \begin{cases} 1, & \text{if } o_{i,C_n} = o_{j,C_n} \ \& \ o_{j,C_n} = \text{label}_j \\ 0, & \text{otherwise.} \end{cases}$$

$$\gamma_{i,j} = \begin{cases} 1, & \text{if } o_{j,C_n} = \text{label}_j \\ 0, & \text{otherwise.} \end{cases}$$

Similar to TM, a higher value of $OTM_{i,j}$ indicates greater similarity between instances i and j.

6 Proposed Measures

For our dynamic class prediction framework, we propose two new distance measures: Probability-Based Template Matching and Probability-Based Template Matching with Accuracy. These measures are described in detail below.

6.1 Probability-Based Template Matching

Probability-Based Template Matching (PTM) maps each data instance into an alternate feature space constructed by using the probability estimates of each classifier in the pool as the values of the features. As previously mentioned, for the two-class problems considered in this paper, the probability estimates are taken with respect to a particular class label, and the choice of label is arbitrary. The similarity between instances i and j, denoted by $PTM_{i,j}$, is calculated as the Euclidean distance between them in this alternate feature space:

$$PTM_{i,j} = \sqrt{\frac{\sum_{n=1}^{N} \left(\phi_{i,j,C_n}\right)^2}{N}}$$

$$\phi_{i,j,C_n} = P_{C_n,i} - P_{C_n,j}$$

where $P_{C_n,i}$ represents the probability estimate in the alternative feature space for data instance i. Similar to the ED, the pair of instances (i, j) with smallest $PTM_{i,j}$ is considered to be most similar.

6.2 Probability-Based Template Matching with Accuracy

Probability-Based Template Matching with Accuracy (PTMA) integrates the correctness of classifiers on validation instance j. We focus on the probability estimates returned by the classifier for the correct class label of the new and validation instances. This gives us four different cases as specified in the equation. To avoid one case eliminating the effect of the other cases, we re-scale all values to be between $[0, 1]$.

$$PTMA_{i,j} = \sqrt{\frac{\sum_{n=1}^{N} (\phi_{i,j,C_n})^2}{N}}$$

$$\phi_{i,j,C_n} = \begin{cases} 2 \mid P_{C_n,i} - P_{C_n,j} \mid, \\ \quad \text{if } o_{i,C_n} = o_{j,C_n} = \text{label}_j \\ \mid 1 - P_{C_n,i} - P_{C_n,j} \mid, \\ \quad \text{if } o_{i,C_n} = o_{j,C_n} \neq \text{label}_j \\ \mid P_{C_n,i} - P_{C_n,j} \mid, \\ \quad \text{if } o_{i,C_n} \neq o_{j,C_n} = \text{label}_j \\ 2 \mid 1 - P_{C_n,i} - P_{C_n,j} \mid, \\ \quad \text{if } o_{i,C_n} \neq o_{j,C_n} \neq \text{label}_j \end{cases}$$

We illustrate how these measures differ with a simple example. Tables 1 and 2 show, respectively, the assigned labels and the corresponding probability estimates for class label l_1 returned by classifiers $C_1, ..., C_5$.

Table 1: Assigned class labels by classifiers

	C_1	C_2	C_3	C_4	C_5	Correct Label
v_1	l_1	l_1	l_2	l_1	l_1	l_1
v_2	l_1	l_1	l_2	l_1	l_2	l_2
v_3	l_2	l_2	l_1	l_1	l_1	l_1

Table 2: Class probabilities (for l_1) by classifiers

	C_1	C_2	C_3	C_4	C_5
v_1	0.53	0.57	0.44	0.90	0.85
v_2	0.92	0.89	0.24	0.52	0.35
v_3	0.46	0.43	0.53	0.88	0.90

We calculate the relevant measures for data instances v_1, v_2, v_3 below:

$$\begin{aligned} TM_{v_1,v_2} &= 0.80, & TM_{v_1,v_3} &= 0.40 \\ OTM_{v_1,v_2} &= 1.00, & OTM_{v_1,v_3} &= 0.66 \\ PTM_{v_1,v_2} &= 0.83, & PTM_{v_1,v_3} &= 0.19 \\ PTMA_{v_1,v_2} &= 0.94, & PTMA_{v_1,v_3} &= 0.11 \end{aligned}$$

We conclude that v_1 and v_2 are more similar than v_1 and v_3, according to both TM and OTM measures. However, v_1 and v_3 are considered more similar instances according to our PTM and PTMA measures. Therefore, considering only the assigned labels to determine similarity may lead to incorrect decisions. This is especially true for instances that are closer to the decision boundary of a classifier, as the classifier provides less confidence in its prediction for those instances.

7 Experimental Setup and Results

In our experiments, we use 13 data sets with varying numbers of features and data instances retrieved from the LIBSVM website (Chang & Lin (2011)). A summary of the data sets and the classifiers generated for each is presented in Table 3.

The programming code was written in MATLAB and LIBSVM (Chang & Lin (2011)) was used to construct RBF kernel SVM classifiers. The training parameters were chosen such that classifiers overfit the data. Each experiment was repeated 100 times, with each run registering a unique seed value for the random number generator.

Table 3: Summary of the data sets and classifiers

Data Set	#Data Points	#Features	%Good Classifiers
a1a	1605	119	100
australian	690	14	64.17
diabetes	768	8	99.74
german	1000	24	100
splice	1000	60	52.77
heart	269	13	59.44
liver disorder	345	6	79.46
sonar	208	60	54.04
breast cancer	683	10	97.14
ionosphere	351	34	88.75
mushrooms	8124	112	54.05
w1a	2477	300	100
rcv	20242	47236	56.65

For each run, the data sets are randomly divided into three subsets such that 60% of the instances is used to train classifiers, 20% is used for validation, and the remaining 20% is used for testing. A pool of 1000 classifiers is then constructed for each data set using a combination of bootstrap instance sampling (as in bagging) and random subspace selection on the training set.[2] In so doing, we ensure that the initial classifier pool is highly diverse. Finally, classifiers with an error rate above 50% are removed from the pool. The last column of Table 3 represents the percentage of the classifiers with less than 50% error rate in the generated pool of classifiers.

In the following sections, we first analyze the static step. In Section 7.2, we perform an experiment to set the value of k for baseline and proposed (dis)similarity measures. Then, we compare the effectiveness of these measures to find the closest k neighbors to make the predictions in Section 7.2. In Section 7.3, we investigate whether the classifier-based feature space can improve the *KNORA* method results. In addition, we choose the appropriate distance measure for the *KNORA* before comparing it with the static step.

Once our comparison for the static step is done, we turn our attention to the dynamic step. We first determine the optimal reduced space size in Section 7.4. In Section 7.5, we examine the contribution of the dynamic step in our framework in addition to the static step, while we compare the dynamic step against common benchmarks in Section 7.6. We then explore various strategies in evaluating the dynamic step in Section 7.7. Finally, in Section 7.8, we compare the performance of the dynamic step against that of the ensembles formed by the classifiers for the reduced space.

7.1 Evaluation of Size of Neighborhood

The number of neighbors considered for each similarity measure is crucial. As a preprocessing step, we perform an experiment in which k is varied over the odd integers from 1 to 25 to find the optimal value. In this step, for each run of a data set, we predict the validation instances using training instances and decide for which k value maximum accuracy is obtained. Table 4 shows the best k value for each similarity measure, averaged over all of the runs. We find that the value of

[2] Due to its size, only 100 classifiers are generated for the *rcv* data set.

k is smaller for the OTM and PTMA measures. A possible explanation is that these two measures take into account the correctness of the decision associated with the validation instances. Consequently, a small neighborhood of validation instances is sufficient to correctly classify the new instance.

Table 4: Best k value for each similarity measure

Data Set	TM	OTM	ED	PTM	PTMA
a1a	9.6	3.3	14.24	9.8	1.42
australian	9.34	1.5	12.14	11.24	1.46
breast_cancer	9.06	1.66	5.14	7.74	1.08
diabetes	11.14	3.22	13.34	12.3	2.66
german	14.52	3.52	14.08	13.68	1.28
heart	7.48	1.56	13.32	7.06	1.86
ionosphere	9.64	2.16	2.84	9.34	1.12
liver_disorder	10.02	2.2	9.02	12.12	3.82
mushrooms	9.56	1	1	10.48	1
rcv	16.5	4.9	1.94	16	3.62
sonar	5.94	1.12	1.86	8.26	1.22
splice	3.46	1.16	10.02	5.48	1
w1a	1.34	1	2.2	1.34	1

7.2 Evaluation of Different Distance Measures

After the best k value is determined for each of the similarity measures presented in Section 5, we perform experiments to assess the accuracy of these measures in classifying new data instances. These experiments use the k-NN algorithm to define the neighborhood for and classify each new instance based on the validation instances deemed to be similar. Table 5 summarizes the results of these experiments: we report mean and standard deviation of accuracy from 100 runs in parentheses, respectively. Entries highlighted in bold in the table indicate the best accuracy achieved for that data set. From these results, it is clear that, on average, PTMA achieves better performance than the other measures.

We also perform pairwise t-tests (at 5% significance level) to compare the proposed distance measures with the baseline measures, with the results tabulated in Table 6. Entries are specified as (x_1, x_2, x_3) which represents (wins,ties,losses) of the proposed methods against the baseline measures over all 13 data sets. For example, when compared to the TM measure, the PTM measure performs statistically significantly better for 6 data sets and worse for 1 data set. For the rest of the data sets, the differences between two measures is not statistically significant. PTM, TM and ED use the same distance measure but in different spaces. As demonstrated by the pairwise t-test results for the PTM measure against the TM and ED measures, the use of the class probability space improves accuracy over either the class prediction space or the original feature space, even without consideration of the classifiers' accuracies for the validation instances. Based on the results for the PTMA measure against the PTM measure, we can conclude that integration of the classifier accuracy into the distance measure further improves accuracy.

Table 6: Pairwise t-test results for dis/similarity measures

	TM	OTM	ED	PTM
PTM	(6,6,1)	(8,2,3)	(10,3,0)	-
PTMA	(7,3,3)	(5,5,3)	(8,3,2)	(6,3,4)

7.3 Comparisons against *KNORA* method

Our proposed framework in Section 3 in both static and dynamic steps finds k nearest neighbors of a new instance. However, unlike *KNORA* instead of using them to form an ensemble of classifiers it uses neighbors' labels to make predictions. In this section, we perform experiments to compare our static step against *KNORA* methods. In addition, *KNORA* uses Euclidean distance measure to find the neighbors of a new instance and the choice of distance measure had no effect on the performance of *KNORA* (Vriesmann et al. (2012)). However, the measures considered by Vriesmann et al. (2012) are based on finding distance in the original feature space. We extend Vriesmann et al. (2012) to analyze the effectiveness of *KNORA* in classifier-based space and find best distance measure for *KNORA* methods before we perform the comparison against static step of our framework.

7.3.1 Evaluating Similarity Measures for KNORA

KNORA-Eliminate and *KNORA-Union* are implemented using the measures discussed in Sections 5 and 6: TM, OTM, ED, PTM, and PTMA. A pairwise t-test at $\alpha = 0.05$ was performed to compare PTM and PTMA against other measures. Entries in tables 7 and 8 represent the number of wins, ties, and losses, respectively, of the proposed measures against the baseline measures (as well as PTMA against PTM) for *KNORA* methods for 13 datasets considered here. For *KNORA-Eliminate* PTMA is always superior to the other measures. However, PTM seems to be more suitable for smaller k values. PTM performs better than other measures for *KNORA-Union*. Interestingly, PTMA performs considerably worse compared to others. As a result of this experiment, we can conclude that $KNORA$ performs well in a classifier based space especially with our proposed measures.

Table 7: Pairwise t-test: comparison of (dis)similarity measures for KNORA-Eliminate

	PTM vs.			*PTMA vs.*			
k	TM	OTM	ED	TM	OTM	ED	PTM
1	(5,7,1)	(6,3,4)	(3,7,3)	(7,1,5)	(5,5,3)	(6,2,5)	(6,3,4)
3	(6,6,1)	(7,4,2)	(5,5,3)	(7,2,4)	(6,4,3)	(8,1,4)	(8,1,4)
5	(5,5,3)	(8,2,3)	(3,9,1)	(7,2,4)	(8,2,3)	(8,2,3)	(7,2,4)
7	(5,5,3)	(7,3,3)	(5,6,2)	(7,3,3)	(7,3,3)	(8,3,2)	(7,3,3)
9	(4,6,3)	(6,4,3)	(6,5,2)	(5,5,3)	(8,2,3)	(9,1,3)	(7,5,1)
11	(4,3,6)	(6,2,5)	(7,4,2)	(5,6,2)	(7,3,3)	(10,1,2)	(9,3,1)
13	(4,3,6)	(5,3,5)	(7,4,2)	(5,5,3)	(6,4,3)	(11,1,1)	(10,2,1)
15	(4,3,6)	(5,2,6)	(7,4,2)	(5,6,2)	(6,3,4)	(11,1,1)	(11,1,1)
17	(4,3,6)	(5,1,7)	(8,4,1)	(6,5,2)	(5,3,5)	(11,1,1)	(11,1,1)
19	(4,4,5)	(5,1,7)	(8,4,1)	(7,5,1)	(5,3,5)	(11,1,1)	(10,2,1)
21	(4,4,5)	(4,2,7)	(9,3,1)	(10,3,0)	(5,3,5)	(11,1,1)	(11,1,1)
23	(2,7,4)	(4,2,7)	(8,4,1)	(10,3,0)	(6,2,5)	(11,1,1)	(11,1,1)
25	(2,7,4)	(4,3,6)	(8,4,1)	(10,3,0)	(7,2,4)	(11,1,1)	(11,2,0)

7.3.2 Static Step vs. *KNORA*

Since the size of an ensemble used to make predictions for each new instance changes for *KNORA*-Eliminate and -Union methods, we analyze the performance of *KNORA* Eliminate and Union against the *static* step of our framework. We explore whether k-NN (Static Step) is better than *KNORA* after neighbors are found by using the proposed measures since the experiments

Table 5: Average accuracy and its standard deviation for dis/similarity measures

Data Set	TM	OTM	ED	PTM	PTMA
a1a	(79.65, 1.34)	(76.18, 10.63)	(78.92, 1.28)	**(81.11, 2.01)**	(77.54, 1.04)
australian	(85.21, 2.84)	(84.2, 5.22)	(85.76, 3.03)	(85.54, 2.93)	**(86.06, 2.70)**
breast_cancer	(96.26, 2.43)	(96.03, 4.64)	(96.13, 1.67)	(96.54, 1.49)	**(96.82, 1.38)**
diabetes	(71.03, 11.74)	(52.8, 20.31)	(72.04, 3.26)	(74.15, 3.87)	**(75.36, 2.50)**
german	(71.35, 1.21)	(71.9, 1.12)	(71.16, 2.07)	**(72.72, 1.79)**	(71.46, 0.92)
heart	(81.96, 5.16)	(82.97, 4.76)	(80.68, 5.36)	(80.83, 5.56)	**(83.08, 5.12)**
ionosphere	(90.93, 12.01)	(82.48, 22.58)	(80.71, 5.93)	(93.74, 2.79)	**(94.04, 2.49)**
liver_disorder	(69.14, 6.15)	**(71.06, 5.71)**	(58.09, 5.71)	(69.14, 5.11)	(70.45, 4.09)
mushrooms	(99.75, 0.09)	(99.67, 0.08)	(98.56, 0.05)	**(100, 0)**	**(100, 0)**
rcv	(86.77, 16.49)	(55.51, 17.08)	(90.54, 0.51)	(96.93, 0.38)	**(97.04, 0.26)**
sonar	(83.16, 5.84)	**(85.83, 5.63)**	(72.15, 7.73)	(82.87, 6.65)	(77.42, 6.54)
splice	(69.05, 7.92)	(66.25, 9.03)	(65.76, 3.82)	**(77.55, 4.84)**	(53.5, 0.96)
w1a	(97.18, 0.27)	(92.91, 0.14)	(97.19, 0.43)	(97.17, 0.41)	(97.13, 0.15)

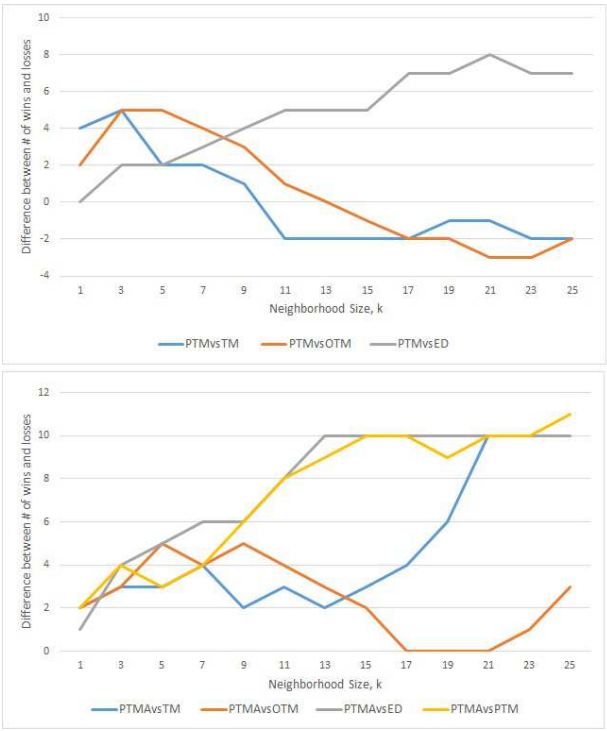


Figure 2: Comparisons of proposed measures against other dis/similarity measures for KNORA Eliminate

in Section 7.3.1 indicate that *KNORA* performs better with PTM and PTMA measures compared to ED, TM, and OTM. Table 9 show the t-test performed to compare *KNORA* and k-NN in classifier-based probability space. Figure 4 illustrates the difference between the number of times *KNORA* outperforms and underperforms against static step of our framework. For instance, for $k = 1$ and distance measure PTM, static step of our framework outperforms *KNORA*-Eliminate for 10 datasets and under-performs for 1 dataset. For 2 datasets, there is a tie. So, for $k = 1$ the Figure 4 shows $1 - 10 = -9$. The results in Table 9 and in Figure 4 show that static step —k-NN with PTM or PTMA— outperforms *KNORA*-Eliminate and -Union with PTM or PTMA.

Table 8: Pairwise t-test: comparison of (dis)similarity measures for KNORA-Union

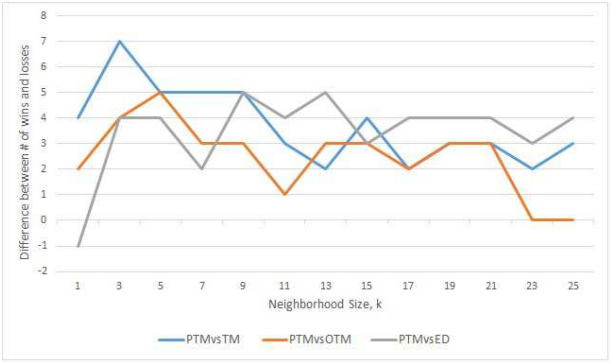
	PTM vs.			*PTMA vs.*			
k	TM	OTM	ED	TM	OTM	ED	PTM
1	(5,7,1)	(6,3,4)	(3,6,4)	(7,1,5)	(5,5,3)	(7,1,5)	(6,3,4)
3	(7,6,0)	(6,5,2)	(5,7,1)	(2,6,5)	(4,5,4)	(2,5,6)	(1,6,6)
5	(5,8,0)	(6,6,1)	(5,7,1)	(2,4,7)	(3,5,5)	(2,6,5)	(2,3,8)
7	(5,8,0)	(4,8,1)	(3,9,1)	(2,4,7)	(2,7,4)	(2,6,5)	(1,5,7)
9	(5,8,0)	(4,8,1)	(6,6,1)	(1,5,7)	(1,8,4)	(2,7,4)	(1,5,7)
11	(3,10,0)	(4,6,3)	(4,9,0)	(2,4,7)	(2,6,5)	(2,5,6)	(1,5,7)
13	(3,9,1)	(5,6,2)	(5,8,0)	(1,5,7)	(1,7,5)	(2,5,6)	(1,5,7)
15	(5,7,1)	(4,8,1)	(4,8,1)	(1,5,7)	(1,7,5)	(2,7,4)	(1,5,7)
17	(3,9,1)	(3,9,1)	(5,7,1)	(1,6,6)	(1,7,5)	(2,8,3)	(1,5,7)
19	(3,10,0)	(4,8,1)	(5,7,1)	(1,7,5)	(1,8,4)	(2,8,3)	(1,6,6)
21	(4,8,1)	(4,8,1)	(5,7,1)	(1,8,4)	(1,8,4)	(2,5,6)	(1,6,6)
23	(3,9,1)	(2,9,2)	(5,6,2)	(1,7,5)	(1,7,5)	(2,5,6)	(1,7,5)
25	(3,10,0)	(1,11,1)	(5,7,1)	(1,9,3)	(1,9,3)	(2,7,4)	(1,7,5)

Table 9: Pairwise t-test: KNORA methods vs. static step

	KNORA-E vs.		*KNORA-U vs.*	
k	PTM-S	PTMA-S	PTM-S	PTMA-S
1	(1,2,10)	(3,3,7)	(1,2,10)	(3,3,7)
3	(1,2,10)	(3,3,7)	(5,4,4)	(3,4,6)
5	(1,1,11)	(3,1,9)	(5,4,4)	(3,3,7)
7	(0,2,11)	(3,2,8)	(5,4,4)	(3,4,6)
9	(0,2,11)	(3,1,9)	(5,3,5)	(3,4,6)
11	(0,1,12)	(3,1,9)	(4,4,5)	(3,5,5)
13	(0,1,12)	(3,1,9)	(5,3,5)	(3,3,7)
15	(0,1,12)	(2,2,9)	(4,4,5)	(3,4,6)
17	(0,1,12)	(2,1,10)	(4,2,7)	(3,4,6)
19	(0,1,12)	(2,1,10)	(4,2,7)	(3,5,5)
21	(0,1,12)	(2,1,10)	(4,2,7)	(3,3,7)
23	(0,1,12)	(2,1,10)	(4,2,7)	(3,2,8)
25	(0,1,12)	(2,1,10)	(4,3,6)	(3,3,7)

7.4 Evaluating Reduced Space Size in Dynamic Step

It was shown that the marginal improvement in the performance of an ensemble diminishes for sizes beyond 25 (Breiman (1996)). In this section, we validate and refine this assumption by investigating the effects of the reduced space size on the performance of our framework. Experiments are run with for reduced space sizes, E, in increments of 5 over the range of 10 to 50, and the results are tabulated in Table 10. Overall, as shown in the table, the choice of the reduce space size has minimal effect on the performance of the framework. Therefore, for the experiments in Sections 7.5–7.7, the size of the reduced space is kept at 25.



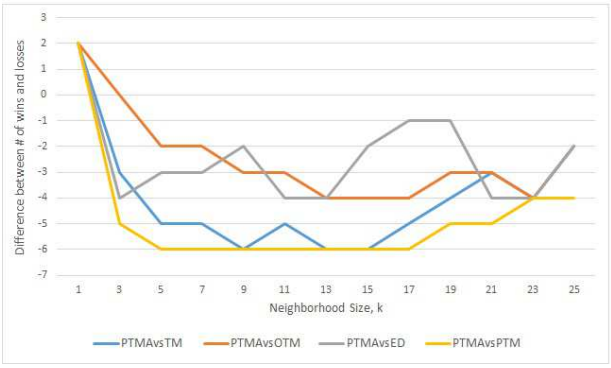


Figure 3: Comparisons of proposed measures against other dis/similarity measures for KNORA Union

Table 10: Pairwise t-test: dynamic vs. static steps across reduced space size (E)

	PTM-D vs.		*PTMA-D vs.*	
E	PTM-S	PTMA-S	PTM-S	PTMA-S
10	(0,7,6)	(3,3,7)	(5,4,4)	(4,6,3)
15	(0,8,5)	(3,3,7)	(5,4,4)	(5,7,1)
20	(0,9,4)	(3,4,6)	(6,3,4)	(5,8,0)
25	(1,8,4)	(3,4,6)	(6,3,4)	(5,8,0)
30	(1,8,4)	(3,3,7)	(7,2,4)	(5,8,0)
35	(2,7,4)	(4,3,6)	(6,3,4)	(5,8,0)
40	(0,9,4)	(4,4,5)	(7,2,4)	(5,8,0)
45	(0,10,3)	(4,3,6)	(6,3,4)	(5,8,0)
50	(1,9,3)	(4,5,4)	(5,4,4)	(4,9,0)

7.5 Comparison of Dynamic and Static Steps

This experiment is performed to evaluate the gain achieved by having the dynamic step as part of our framework. Even though the running time of our system is dominated by the static step, for the dynamic step the neighborhood of a new instance is refined using the information retrieved from the static step to form the "Reduced Space" and a new set of neighbors is found. These calculations require some additional time cost to the system.

Tables 11 and 12 compare the results from the static and dynamic steps for both the PTM and PTMA distance measures. In Table 11, we report mean and standard deviation of accuracy from 100 runs in parentheses, respectively. As observed in these two tables, the dynamic step for the PTM and PTMA measures (hereafter referred to as PTM-D and PTMA-D, respectively) consistently outperforms the static step for the ED measure, even for data sets where it underperforms the static step for the PTM and PTMA measures

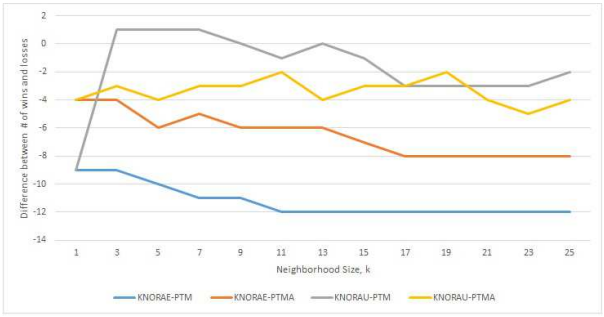


Figure 4: Comparison of Static Step with KNORA methods

(termed PTM-S and PTMA-S). Further, an important result is that PTMA-D performs at least as well as PTM-S and PTMA-S. However, PTM-D performs significantly worse than PTM-S and PTMA-S. When the classifiers for the reduced space are chosen, it is aimed to find expert classifiers on the neighbor instances and the results from this experiment clearly shows that expert classifiers can be identified by considering not just the confidence but also the accuracy of the classifiers on the neighbors which explains the change in the performance of PTMA-D and PTM-D.

Table 12: Pairwise t-test: dynamic vs. static steps

	ED	PTM-S	PTMA-S
PTM-D	(8,4,1)	(1,8,4)	(3,4,6)
PTMA-D	(10,2,1)	(6,3,4)	(5,8,0)

7.6 Dynamic Step vs. Common Benchmarks

We also compared the performance obtained with our framework against the common benchmarks. The baseline benchmarks considered here are:

- Use of the best classifier from set $\mathcal{C}$;
- Use of the 25 best-performing classifiers from set $\mathcal{C}$ with majority voting
- Use of a single SVM classifier trained on all of the training data.

In addition, we used "Modified Best Improvement" algorithm to find the best ensemble of size E over the validation set. Since it can be impractical to evaluate all $\binom{N}{E}$ possible ensembles, as required for the traditional "Best Improvement" algorithm, a modified form of the algorithm is employed: given one of the E classifier slots in the ensemble, each of the $N - E$ unused classifiers is, in turn, substituted into that slot, and the accuracy of the ensemble is re-evaluated. The classifier that provides the best ensemble performance is permanently assigned to the slot, and the displaced classifier is returned to the pool of unused classifiers. This process is then repeated for the next classifier slot in the ensemble until all slots have been processed. We perform an experiment in which E is set to 25 to find the best ensemble on the validation set for a given data set. Tables 13 and 14 compare the performance of our proposed measures against these benchmarks in terms

Table 11: Average accuracy and its standard deviation in the static and dynamic steps

Data Set	ED	PTM-S	PTMA-S	PTM-D	PTMA-D
a1a	(78.92, 1.28)	**(81.11, 2.01)**	(77.54, 1.04)	(79.37, 2.08)	(79.69, 1.63)
australian	(85.76, 3.03)	(85.54, 2.93)	**(86.06, 2.7)**	(85.43, 2.79)	(85.79, 2.81)
breast_cancer	(96.13, 1.67)	(96.54, 1.49)	**(96.82, 1.38)**	(96.63, 1.49)	**(96.82, 1.37)**
diabetes	(72.04, 3.26)	(74.15, 3.87)	(75.36, 2.5)	(74.39, 3.57)	**(75.51, 2.58)**
german	(71.16, 2.07)	**(72.72, 1.79)**	(71.46, 0.92)	(71.65, 1.87)	(71.89, 1.26)
heart	(80.68, 5.36)	(80.83, 5.56)	**(83.08, 5.12)**	(80.33, 5.45)	(82.95, 4.92)
ionosphere	(80.71, 5.93)	(93.74, 2.79)	(94.04, 2.49)	**(94.13, 2.64)**	**(94.13, 2.47)**
liver_disorder	(58.09, 5.71)	(69.14, 5.11)	**(70.45, 4.09)**	(68.62, 5.71)	**(70.49, 4.52)**
mushrooms	(98.56, 0.05)	**(100, 0)**	**(100,0)**	**(100, 0)**	**(100, 0)**
rcv	(90.54, 0.51)	(96.93, 0.38)	**(97.04, 0.26)**	(96.93, 0.37)	**(97.03, 0.26)**
sonar	(72.15, 7.7)	**(82.87, 6.65)**	(77.42, 6.5)	(82.68, 5.82)	(80.77, 6.15)
splice	(65.76, 3.82)	**(77.55, 4.84)**	(53.5, 0.96)	(75.85, 4.06)	(55.22, 1.2)
w1a	(97.19, 0.43)	(97.17, 0.41)	(97.13, 0.15)	(96.79, 0.58)	**(97.19, 0.24)**

of average accuracy and pairwise t-test results, respectively. From these tables, we conclude that PTMA outperforms all of the benchmarks. Even PTM, a basic distance measure in the probability space, performs better than any of these benchmarks.

Table 14: Pairwise t-test: dynamic step vs. benchmark methods

	Best Classifier	Best 25	Single SVM	Best Ens
PTM-D	(12,0,1)	(7,4,2)	(8,2,3)	(4,7,2)
PTMA-D	(12,1,0)	(9,3,1)	(8,3,2)	(9,3,1)

7.7 Evaluating Different Strategies for Dynamic Step

As described in the previous section, our proposed framework takes into account the class prediction probabilities assigned to the k-nearest neighbors found in the static step to choose the classifiers whose outputs will be used for generating the reduced probability space for the dynamic step. In this section, we investigate different strategies for choosing a subset of classifiers from the original pool to be used in the dynamic step. In addition to the selection strategy presented earlier, five alternate strategies are also implemented:

- **Local Accuracy (LA):** Classifiers which correctly classify at least one of the k neighbors are added to $\mathcal{C}'$. If $|\mathcal{C}'| > 25$, then we consider only the 25 top-performing classifiers.
- **LA $+\mathbf{Acc_{Val}}$:** This strategy is similar to LA; however, if $|\mathcal{C}'| < 25$, additional classifiers are selected to reach a size of 25. These classifiers are chosen from the best-performing classifiers on the validation instances that are not already in $\mathcal{C}'$.
- **Conditional LA (C + LA):** For this strategy, we only consider the classifiers with $\geq 50\%$ accuracy in the neighborhood. If there is no such classifier found, then the label for a new instance is assigned randomly. If there are more than 25 classifiers, then we consider more accurate 25 classifiers in the region.
- **LA + Closeness (LA + Cl):** If the total number of classifiers which correctly classify at least one of the k neighbors is less than 25, we also add the classifiers which are close to making the correct decision on the neighbors. Closeness is defined as the absolute difference between the probability estimate returned by the classifier for the correct class label and the 0.5 decision boundary.
- **Minimum Distance (MinDist):** This strategy chooses classifiers that minimize the average distance between a test instance and its neighbors.

Table 15: Pairwise t-test: dynamic vs. static steps under different strategies

	PTM-D vs.		*PTMA-D vs.*	
Version	PTM-S	PTMA-S	PTM-S	PTMA-S
LA	(1,7,5)	(4,4,5)	(4,5,4)	(2,6,5)
LA+Acc_{Val}	(0,6,7)	(4,2,7)	(2,6,5)	(1,7,5)
C+LA	(0,6,7)	(4,2,7)	(3,6,4)	(1,7,5)
LA+Cl	(0,6,7)	(4,2,7)	(2,6,5)	(1,7,5)
MinDist	(0,6,7)	(4,2,7)	(3,5,5)	(1,7,5)

Table 15 lists the pairwise t-test results for each of these classifier selection strategies against the static step of our framework. As observed in the table, none of these strategies yield improved accuracy over the static step. In contrast, the selection strategy discussed in Section 7.5 does improve performance over the static step when the PTMA measure is used. Intuitively, this can be understood because of the complementary natures of the classifier selection method, which favors *high* levels of confidence, and the similarity measure, which favors *similar* levels of confidence. Further, while the PTM and PTMA measures achieve improved performance by considering instance classification as a continuous range of probabilities rather than as discrete labels–thereby being able to recognize the similarity of two instances close to but on opposite sides of the decision boundary–the area near the decision boundary still reflects uncertainty regarding the nature of the new instance. Therefore, especially refining the static step with PTMA measure to favor high classification confidence on the neighbors while still considering the full range of class label probabilities, as is done in the dynamic step, further improves the accuracy of our framework. However, none of the strategies proposed in this section removes the uncertainty associated with the decision boundary, and therefore they do not offer improved performance.

7.8 Dynamic Step vs. Classifiers in the Reduced Space

The reduced space technique of our framework can be considered a dynamic ensemble selection method. In-

Table 13: Average accuracy and its standard deviation for benchmark methods and dynamic step

Data Set	Best Classifier	Best 25	Single SVM	Best Ens	PTM-D	PTMA-D
a1a	(75.50, 2.33)	(78.06, 1.05)	(75.40, 0.78)	(77.11, 1.10)	(79.37, 2.08)	**(79.69, 1.63)**
australian	(75.93, 6.64)	(84.95, 3.03)	(82.07, 2.94)	(85.33, 2.87)	(85.43, 2.79)	**(85.79, 2.81)**
breast_cancer	(94.80, 2.19)	(96.45, 1.40)	(95.80, 1.40)	(96.50, 1.47)	(96.63, 1.49)	**(96.82, 1.37)**
diabetes	(65.94, 2.05)	(65.12, 0.38)	(73.95, 2.66)	(75.01, 2.90)	(74.39, 3.57)	**(75.51, 2.58)**
german	(69.89, 0.6)	(70.01, 0.01)	(70.37, 1.22)	(71.31, 1.15)	(71.65, 1.87)	**(71.89, 1.26)**
heart	(72.96, 6.76)	(80.85, 5.30)	(73.28, 5.71)	(80.89, 5.21)	(80.33, 5.45)	**(82.95, 4.92)**
ionosphere	(91.80, 3.63)	(93.27, 3.03)	(92.19, 3.09)	(93.21, 2.84)	**(94.13, 2.64)**	**(94.13, 2.47)**
liver_disorder	(63.03, 6.25)	(64.22, 3.86)	**(71.13, 4.71)**	(68.57, 4.28)	(68.62, 5.71)	(70.49, 4.52)
mushrooms	(99.76, 0.62)	(99.97, 0.51)	(99.99, 0.01)	(99.99,0.01)	**(100, 0.0)**	**(100, 0.0)**
rcv	(95.17, 1.61)	(97.03, 0.26)	**(97.17, 0.25)**	(97.05, 0.24)	(96.93, 0.37)	(97.03, 0.26)
sonar	(72.28, 7.73)	(79.54, 6.81)	(73.85, 6.26)	(80.71, 6.50)	**(82.68, 5.82)**	(80.77, 6.15)
splice	(55.37, 3.61)	(56.68, 1.44)	(56.185, 1.22)	(56.83, 1.98)	**(75.85, 4.06)**	(55.22, 1.22)
w1a	(97.10, 0.18)	(97.11, 0.12)	(97.16, 0.25)	(97.13, 0.16)	(96.79, 0.58)	**(97.19, 0.24)**

stead of performing k-NN in this reduced space, those classifiers can be used to form an ensemble to make predictions on the new instance. We performed this experiment with several ensemble and neighborhood size values, (E and k) respectively. As mentioned in Section 7.7, we employed different strategies to choose the classifiers to form the reduced space (i.e. the *dynamic* ensemble). Figures 5 - 8 are plotted based on the difference between the number of wins and losses of dynamic step of our framework against dynamic ensemble selection strategy for a given (k, E) pair.

7.8.1 Dynamic Step vs. *Max-Confidence* Ensemble

Figures 5 and 6 summarize the comparison results between dynamic framework (with PTM/PTMA) against classifiers chosen to form the reduced space per their confidence on the k neighbors chosen in the static step. Figure 5 show that for PTM measure if $k > 7$ and $E > 11$ our dynamic framework performs at least as well as using the E classifiers that form the reduced space. According to Figure 6, Dynamic Framework is still better than the ensemble built by using the classifiers that are chosen to form the reduced space based on their high confidence on the neighbors of a test instance. Dynamic framework with PTMA does not dominate the ensemble as strongly as PTM does but it almost always performs at least as well as the ensemble.

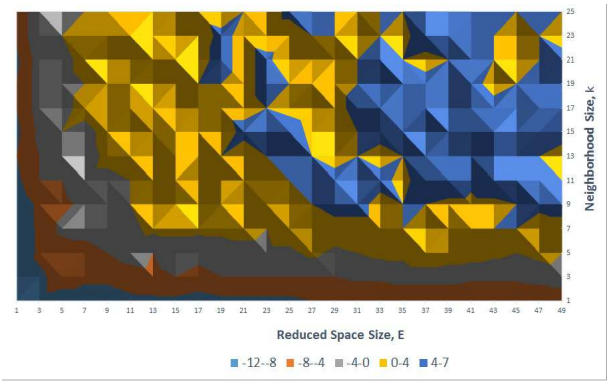


Figure 5: Comparison of Dynamic Step (PTM-D) against the ensemble formed using the classifiers in the reduced space per MaxConf strategy

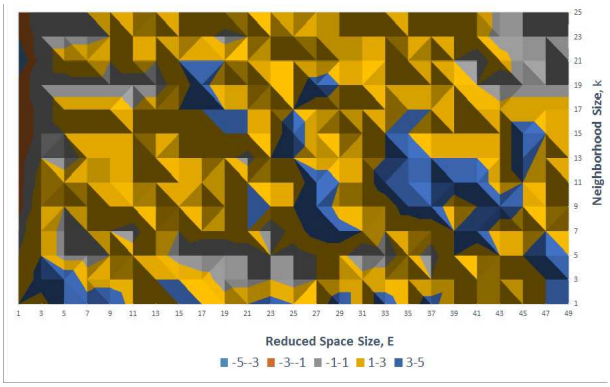


Figure 6: Comparison of Dynamic Step (PTMA-D) against the ensemble formed using the classifiers in the reduced space per MaxConf strategy

7.8.2 Dynamic Step vs Local Accuracy Based Selection Methods

The local accuracy based classifier selection methods that are used for reducing the space in the dynamic step of our framework performed similarly. We present results for the LA method in Figures 7 and 8. PTM-D performs worse or similar to dynamic ensemble generated based on local accuracy. However, PTMA-D performs particularly better for smaller k values. Change in ensemble size seems to have no effect in the performance of PTMA-D.

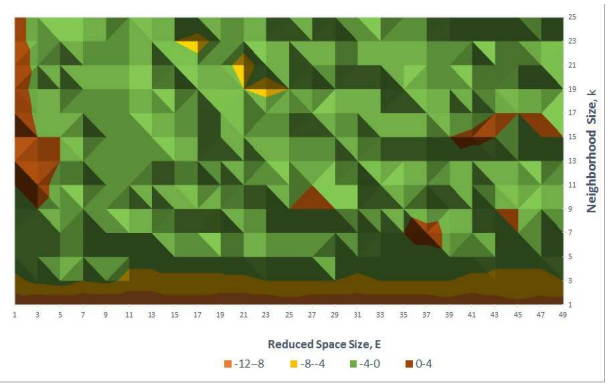


Figure 7: Comparison of Dynamic Step (PTM-D) against the ensemble formed using the classifiers in the reduced space per LA strategy

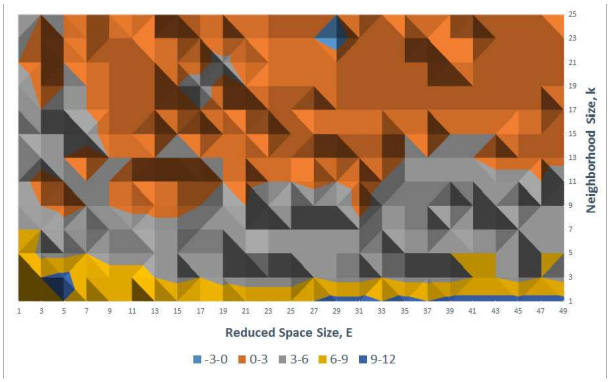


Figure 8: Comparison of Dynamic Step (PTMA-D) with the ensemble formed using the classifiers in the reduced space per LA strategy

8 Conclusion and Future Work

This work proposes a framework for dynamic class prediction using two distance measures (PTM and PTMA) defined based on the probability estimates returned for a particular class label by classifiers for data instances. These measures are compared to three baseline (dis)similarity measures. From our experiments, we conclude that the two proposed measures outperform the baseline measures. Additionally, our experiments reveal that using classifiers' outputs is better than using original feature space to find similar instances. We then compare our results from the static and dynamic steps to some of the traditional classifier and ensemble selection methods. We conclude that k-NN in the probability-estimate-based classifier space outperforms the best classifier, the top performing 25 classifiers, an SVM classifier trained using the entire training data set, and the best ensemble (of size 25) on the validation set.

We considered stationary data for the experiments. This framework can easily be generalized to streaming data as new coming data can be buffered. The size of the pool of classifiers can be controlled by replacing the classifier that performs worst on the buffered instances with a classifier trained on these data instances. When a new classifier is trained, validation set can also be updated. The closest validation instances to the buffered instances can be replaced by these new instances if they also have the same labels. Otherwise, the new instances add new information to the system and should be added to the system without removing any other data instances. As a future work, we would like to modify our framework to account for data streams.

In Section 6, PTM and PTMA measures are defined for two class problems. We would also like to consider multi-class problems as an extension. Multi-class problems present several challenges. Most importantly, unlike two-class problems, there is no clear decision boundary in the probability space. Nonetheless, we believe that the measures presented in this paper could easily be modified to investigate multi-class problems. For example, a simple way to overcome this obstacle would be to transform these problems into a set of two-class problems and training base classifiers accordingly. Alternatively, we could instead consider the probability for each class label of each classifier as an orthogonal dimension of the probability space and calculate the Euclidean distance in this space. However, this approach increases the size of the space by a factor equal to the number of class labels, which makes the distance between two instances less meaningful.

References

Breiman, L. (1996), 'Bagging predictors', *Machine Learning* **24**(2), 123–140.

Cavalin, P., Sabourin, R. & Suen, C. (2010), Dynamic selection of ensembles of classifiers using contextual information, *in* N. Gayar, J. Kittler & F. Roli, eds, 'Multiple Classifier Systems', Vol. 5997 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, pp. 145–154.

Chang, C.-C. & Lin, C.-J. (2011), 'LIBSVM: A library for support vector machines', *ACM Transactions on Intelligent Systems and Technology (TIST)* **2**(3), 27.

Didaci, L. & Giacinto, G. (2004), Dynamic classifier selection by adaptive K-nearest-neighbourhood rule, *in* 'Multiple Classifier Systems', Springer, pp. 174–183.

Didaci, L., Giacinto, G., Roli, F. & Marcialis, G. L. (2005), 'A study on the performances of dynamic classifier selection based on local accuracy estimation', *Pattern Recognition* **38**(11), 2188–2191.

Dos Santos, E. M., Sabourin, R. & Maupin, P. (2008), 'A dynamic overproduce-and-choose strategy for the selection of classifier ensembles', *Pattern Recognition* **41**(10), 2993–3009.

Freund, Y. & Schapire, R. E. (1996), Experiments with a new boosting algorithm, *in* 'International Workshop on Machine Learning', Vol. 96, Morgan Kaufmann, pp. 148–156.

Giacinto, G. & Roli, F. (2001), 'Dynamic classifier selection based on multiple classifier behaviour', *Pattern Recognition* **34**(9), 1879–1882.

Ho, T. K. (1998), 'The random subspace method for constructing decision forests', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **20**(8), 832–844.

Ko, A. H., Sabourin, R. & Britto Jr, A. S. (2008), 'From dynamic classifier selection to dynamic ensemble selection', *Pattern Recognition* **41**(5), 1718–1731.

Menahem, E., Rokach, L. & Elovici, Y. (2009), 'Troika–an improved stacking schema for classification tasks', *Information Sciences* **179**(24), 4097–4122.

Ting, K. M. & Witten, I. H. (1997), Stacking bagged and dagged models, *in* 'ICML', pp. 367–375.

Tumer, K. & Ghosh, J. (1996), 'Error correlation and error reduction in ensemble classifiers', *Connection Science* **8**, 385–404.

Vriesmann, L. M., Jr, A. D. S. B., Oliveira, L. E. S. D., Sabourin, R. & Ko, A. H.-R. (2012), 'Improving a dynamic ensemble selection method based on oracle information', *International Journal of Innovative Computing and Applications* **4**(3), 184–200.

Wolpert, D. H. (1992), 'Stacked generalization', *Neural networks* **5**(2), 241–259.

Woods, K., Kegelmeyer Jr, W. P. & Bowyer, K. (1997), 'Combination of multiple classifiers using local accuracy estimates', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **19**(4), 405–410.

Zhang, Y., Burer, S. & Street, W. N. (2006), 'Ensemble pruning via semi-definite programming', *The Journal of Machine Learning Research* **7**, 1315–1338.